Docket No. 67115
20000183

USING NEURAL NETWORKS FOR DATA MINING

CROSS-REFERENCE TO RELATED APPLICATIONS

This application claims the benefit of the following co-pending U.S. provisional applications:

(a) Serial No. 60/374,064, filed April 19, 2002 and entitled "PROCESSING MIXED NUMERIC AND/OR NON-NUMERIC DATA";

(b) Serial No. 60/374,020, filed April 19, 2002 and entitled "AUTOMATIC NEURAL-NET MODEL GENERATION AND MAINTENANCE";

(c) Serial No. 60/374,024, filed April 19, 2002 and entitled "VIEWING MULTI-DIMENSIONAL DATA THROUGH HIERARCHICAL VISUALIZATION";

(d) Serial No. 60/374,041, filed April 19, 2002 and entitled "METHOD AND APPARATUS FOR DISCOVERING EVOLUTIONARY CHANGES WITHIN A SYSTEM";

(e) Serial No. 60/373,977, filed April 19, 2002 and entitled "AUTOMATIC MODEL MAINTENANCE THROUGH LOCAL NETS"; and

(f) Serial No. 60/373,780, filed April 19, 2002 and entitled "USING NEURAL NETWORKS FOR DATA MINING".

TECHNICAL FIELD

This application relates to data mining. In particular, the application relates to using neural nets and other artificial intelligence techniques for data

mining.

DESCRIPTION OF RELATED ART

As use of computers and other information and communication appliances proliferate in the current information age, data, numeric as well as non-numeric (for example, textual), frequently is collected from numerous sources, such as the Internet. Further, large amounts of data exist in many databases. Much of the data is collected for archiving purposes only and therefore, in many instances, are stored without organization. Sifting through the morass of data to extract useful information for a specific purpose may be a substantial challenge.

For example, business concerns are finding an increasing need, in order to remain competitive in their business market, to effectively analyze and extract useful information from data they and/or others have collected and use the extracted information to improve operation of the business. This, however, often may be a daunting task.

Data mining is the analysis of large qualities of data in order to extract useful information from the data, such as for making predictions over new data (also called predictive analysis). A number of data mining products are available. However, current commercial products which allow data mining of the wealth of information on the Web

require the client application to maintain a predictive model, although a service broker may collect or store raw data and forward it to the client upon demand. Since the client must maintain the predictive model, the resources of the client machine may be overwhelmed when the application is executed.

SUMMARY

This application provides a data mining system. In one embodiment, the data mining system includes a client and a service broker configured to include an interface to receive a consultation request from the client. The service broker forwards the consultation request to a Neugent to invoke a consultation of the Neugent. After the Neugent is consulted, the service broker forwards to the client a result object returned by the Neugent.

The service broker also may include a training interface, and receives through the training interface a training request from the client, the training request including training data. The service broker forwards the training request including the training data to the Neugent to invoke training of the Neugent with the training data. The training request may include a parameter specifying a ratio to split the training data between training the Neugent and testing or validating the Neugent. The service broker may forward to the client a training result object returned by the Neugent after training of the Neugent.

The application also provides a method for providing to a remote client machine a service to consult a Neugent. In one embodiment, the method includes receiving a consultation request from the remote client machine, forwarding the consultation request to the Neugent to

invoke a consultation of the Neugent, and forwarding to the remote client machine a result object returned by the Neugent.

The application also provides a method for providing to a remote client machine a service to train a Neugent. According to one embodiment, the method includes receiving a train request from the remote client machine, forwarding the train request to the Neugent to invoke training of the Neugent, and forwarding to the remote client machine a training result object returned by the Neugent.

BRIEF DESCRIPTION OF THE DRAWINGS

The features of the present application can be more readily understood from the following detailed description with reference to the accompanying drawings wherein:

FIG. 1A shows a block diagram of a data mining system, according to one embodiment of the present disclosure;

FIG. 1B shows a schematic view of a data mining system, according to another embodiment;

FIG. 2A shows a flow chart of a method for providing to a remote client machine a service to consult a Neugent, according to one embodiment;

FIG. 2B shows a flow chart of a method for providing to a remote client machine a service to train a Neugent, according to one embodiment;

FIG. 3 shows a schematic view of a functional-link net structure;

FIGS. 4A and 4B show class diagrams for web services interface methods of Value Predict Neugent, according to one embodiment;

FIGS. 5A, 5C, 5E, 5G and 5I show object schemas for assorted Neugents classes, according to another embodiment; and

FIGS. 5B, 5D, 5F, 5H, 5J and 5K show class diagrams for the web service interface of the Neugents classes;

FIGS. 6A, 6C, 6E, 6G and 6I show object schemas for assorted Neugents classes, according to a third embodiment;

FIGS. 6B, 6D, 6F, 6H, 6J and 6K show class diagrams for the web service interface of the Neugents classes, according to the third embodiment;

FIGS. 7A through 7F show class diagrams for web service interface of assorted Neugents classes, according to a fourth embodiment;

FIG. 7G shows an object schema for the Value Predict Neugent, according to the fourth embodiment;

FIGS. 8A, 8D, 8F, 8H and 8J show object schemas for assorted Neugents classes, according to a fifth embodiment;

FIGS. 8B, 8C, 8E, 8G, 8I and 8K show class diagrams for the web service interface of the Neugents classes, according to the fifth embodiment;

FIG. 9A shows an object schema for Value Predict Neugent, according to a sixth embodiment;

FIGS. 9B and 9C show class diagrams for the web service interface of the Value Predict Neugent, according to the sixth embodiment;

FIGS. 10A and 10C through 10F show class diagrams for the web service interface of assorted Neugents classes, according to a seventh embodiment; and

FIG. 10B shows an object schema for Value Predict Neugent, according to the seventh embodiment.

DETAILED DESCRIPTION

This application provides tools (in the form of systems and methodologies) for data mining. For example, the tools may include one or more computer programs or software modules stored on a conventional program storage device or computer readable medium, and/or transmitted via a computer network or other transmission medium.

A data mining system, according to a client-server paradigm, is explained below with reference to FIG. 1A. It should be understood, however, that the tools of the present application are not limited to a client-server programming model, and may be adapted for use in peer-to-peer systems, message passing systems, as well as other programming models.

A data mining system 10 includes a client 11, one or more Neugents 13, and a service broker 15. The service broker 15 may be configured as a server, and includes an interface to receive a consultation request from the client. The service broker may also receive a train request from the client, and typically is (although it need not be) a remote server. Neugents 13 are further described below.

A method for providing to a remote client machine a service to consult a Neugent, in accordance with one embodiment, is described with reference to FIGS. 1A and 2A.

After the service broker 15 receives a consultation request from the remote client machine (step S21), the service broker forwards the consultation request to a Neugent 13 to invoke a consultation of the Neugent (step S22). After the Neugent 13 is consulted, the service broker 15 forwards to the client a result object returned by the Neugent (step S23).

The consultation request, according to one embodiment, includes data for consulting a Neugent 13. The Neugent 13 performs a predictive analysis of the data included in the consultation request.

According to another embodiment, the consultation request includes identification of a source of data for consulting a Neugent 13. The Neugent 13 performs a predictive analysis of input data obtained from the source identified in the consultation request.

According to another embodiment, the service broker 15 is a remote server. The consultation request from the client 11 to the remote server may include an Extended Markup Language document. The Neugent may be server-side.

A method for providing to a remote client machine a service to train a Neugent, according to one embodiment, is described with reference to FIGS. 1A and 2B. After the service broker 15 receives a train request from the remote client machine (step S26), the service broker forwards the

train request to a Neugent 15 to invoke training of the Neugent (step S27). After the Neugent is trained, the service broker forwards to the client a training result object returned by the Neugent (step S28).

A Neugent may group training data patterns into clusters, with each cluster corresponding to a group of similar data patterns, and predict a probability of membership of an input pattern to a selected group.

A Neugent may group training non-numeric (for example, textual) patterns into clusters, with each cluster corresponding to a group of similar non-numeric patterns, and predict a probability of membership of an input non-numeric pattern to a selected group.

A Neugent may form a cluster model by grouping training data patterns into a plurality of clusters, with each cluster corresponding to a group of similar data patterns, and determining for each cluster probabilities of transition from the cluster to each of the other clusters. The Neugent predicts a probability of an event occurring by applying an input pattern to the cluster model.

A Neugent may form an input-output model associated with a set of training data patterns, and predict an output value by applying the model to an input pattern. The Neugent may include a functional-link net.

A Neugent may form rules associated with corresponding

relationships in a set of training data patterns, and predict an outcome by applying the rules to an input pattern.

Neugents technologies include assorted methodologies for recognizing patterns in data and for using those patterns to make predictions on new data. New data is analyzed to determine the pattern into which it falls, thereby providing a prediction of future behavior based on the behavior that has characterized the pattern in the past.

One group of underlying methodologies is often referred as neural net technology. A neural net is a weighted network of interconnected input/output nodes. Neugent technology covers a broader range of pattern recognition methodologies, in addition to neural net models.

For example, Neugents may include ClusteringNeugent, DecisionNeugent, EventPredictNeugent, TextClusteringNeugent and ValuePredictNeugent model methodologies.

ClusteringNeugent uses a cluster model methodology which groups patterns that are alike, and predicts the probability of membership to a specific group.

DecisionNeugent uses a decision tree model methodology which uncovers rules and relationships in data, formulates rules to describe those relationships, and predicts

outcomes based upon the discovered rules.

EventPredictNeugent uses a cluster model methodology with transition calculation to predict the probability of an event occurring.

TextClusteringNeugent uses a cluster model methodology which groups training data patterns comprising textual (or non-numeric) material that are alike, and predicts a probability that specified textual (or non-numeric) data with which the model is consulted is a member of (or belongs to) a specific group.

ValuePredictNeugent uses a functional-link neural net model methodology to predict the value of a variable (or values for a set of variables).

A functional-link net is one type of neural net which can be used to model a functional relationship between input and output. A functional-link net may be used to approximate any scalar function with a vector of inputs, $\boldsymbol{x}$, and an output y, and therefore is a universal approximator. The structure of a functional-link net with non-linearity fully contained in a functional-link layer is illustrated in FIG. 3. The nodes in the functional-link layer have associated non-linear basis functions. Since non-linearity is fully contained in the functional-link layer, and the rest of the net may be linear, linear training techniques such as regression-based training may be used with a

functional-link net structure. Linear training refers to techniques that solves the parameters in the net through linear algebra techniques. Examples of functional-link net methodologies are described in commonly owned U.S. Patents Nos. 4,979,126, 5,734,796, 6,134,537 and 6,212,509 which are incorporated herein in their entirety by reference.

Some methodologies associated with EventPredictNeugent are described in commonly-owned U.S. Patent No. 6,327,550 which is incorporated herein by reference.

Additional clustering, neural net, decision tree and other predictive modeling methodologies are described in the following commonly-owned U.S. Patent Applications, which are also incorporated herein by reference:

Serial No. 60/374,064, filed April 19, 2002 and entitled PROCESSING MIXED NUMERIC AND/OR NON-NUMERIC DATA;

Serial No. 60/374,020, filed April 19, 2002 and entitled AUTOMATIC NEURAL-NET MODEL GENERATION AND MAINTENANCE;

Serial No. 60/374,024, filed April 19, 2002 and entitled VIEWING MULTI-DIMENSIONAL DATA THROUGH HIERARCHICAL VISUALIZATION;

Serial No. 60/374,041, filed April 19, 2002 and entitled METHOD AND APPARATUS FOR DISCOVERING EVOLUTIONARY CHANGES WITHIN A SYSTEM;

Serial No. 60/373,977, filed April 19, 2002 and

entitled AUTOMATIC MODEL MAINTENANCE THROUGH LOCAL NETS; and

Serial No. 60/373,780, filed April 19, 2002 and entitled "USING NEURAL NETWORKS FOR DATA MINING".

Each Neugent provides the following methods, which are commonly referred to collectively as an "Application Programmer Interface", or "API", and referred to in connection with Web services simply as "services".

Train is a process of providing data (also referred to more specifically as training data patterns) to a Neugent so that the Neugent performs statistical or other data analysis of the training data patters which provides the basis for future predictions. The output of training a Neugent is a model or other data classification mechanism, which becomes the means by which the Neugent recognizes patterns.

Consult is a process of providing new data to a Neugent (also referred to as data for consulting the Neugent) so that the Neugent uses its model, as developed during training, to provide a prediction from the new data.

A Web service enabled implementation of the train and consult methods of the Neugents, according to an exemplary embodiment, is described below, with reference to FIGS. 1B and 5A through 10F. The train and consult methods are made available to client programs through Web services

technology. Typically, only data may be passed between a client and a Neugent. Accordingly, the methodologies described in this disclosure place no burden on the client to maintain a predictive model. The complexity of client/server interfaces may be reduced by simplifying protocols and by hiding issues (for example, making them transparent to the user) of platform technology mismatches.

For example, Web services technology may be based on invoking procedures in a remote server (also referred herein as "Web Service Broker" or "WSB"), such as by transmitting an Extended Mark-up Language (XML) document, which is a text document, over the HTTP protocol, as depicted in FIG. 1B. In order for Web Service Broker 45 to invoke the train and consult methods of a Neugent 43, the structure of the XML documents calling the corresponding methods of the Neugent is precisely specified. The training and consultation API of the Neugents preferably is rigorously defined so that they can be invoked by the WSB. In addition, an interface is implemented within each respective Neugents.

Each of the Neugents mentioned above defines its own specification for training and consulting services (see, for example, FIGS. 4A-10F). The common elements of each Neugent interface include input data, train result and consult result.

For both the train and consult services, a collection of data is passed to the Neugent. Data passed to the train service and the consult service may be referred to as training data (also referred herein as "trainData") or consultation data (also referred herein as "consultData"), respectively. In some cases (for example, the ValuePredictNeugent), additional parameters may be passed when training the Neugent, such as to determine the percentage of the training data split between training the model and validating or testing the model. The Neugents typically use numeric data as input. However, the TextClusteringNeugent also accommodates textual (or other non-numeric) data and the DecisionNeugent accommodates alpha-numeric data.

Except for EventPredictNeugent, each Neugent returns an object as a result of a training session. The object provides information about the result of the training session. For ValuePredictNeugent, an object representing the Neugent may be returned as part of the structure of the train result.

For each Neugent type, the Neugent returns an object as a result of a consultation. Neugents may differ, however, with regard to a structure of the consultation return object. See, for example, FIGS. 5A-5K, in which only the TextClusteringNeugent and the ClusteringNeugent

return similarly structured objects. The ValuePredictNeugent may return the ValuePredictNeugent object itself as part of the returned consultation object.

The specification of Neugents train and consult services may be mapped to the architecture of the Neugent class (discussed below).

The WSB API Interface is discussed exemplarily below for the ValuePredictNeugent only.

The WSB API can include a number of classes, with the ValuePredictNeugent class including train and consult methods.

For example, the ValuePredictNeugent class may include the following train and consult methods: ValueNeugentTrainResult train(Collection of Pattern trainData, Double validationPercentage, Boolean returnResultFlag); and ValueNeugentConsultResult consult(Collection of Pattern consultData).

The user sets up a collection of data under the Pattern class. The Pattern class is a container for a row of data passed to the train or consult method. After passing the data collection into the train or consult method, a ValueNeugentTrainResult object, or a ValueNeugentConsultResult object is returned.

The ValueNeugentTrainResult class contains the results from the ValuePredictNeugent train method, and may include

the following fields (FIG. 4A):

trainStatus indicates a process status when it returns, and is checked in order to determine if the train method returns successful;

modelTrainError indicates an overall training error of a model (for all model outputs);

modelValidationError indicates an overall validation error of the model (for all model outputs);

numberOfData indicates a number of patterns used for training;

trainError indicates for each output in the OFldNList property of the Neugent instance a corresponding training error;

validationError is validation error for each individual target in OFldNList and is the same as modelValidationError when there is only one output;

trainQualityScore indicates for each output in the OFldNList property of the Neugent instance a corresponding training quality score;

validationQualityScore indicates for each output in the OFldNList property of the Neugent instance a validation quality score;

trainResult is a collection consisting of pattern label and model predict values of each target for each pattern;

validationResult is an inner collection consisting of pattern label and model predict values of each target for each pattern;

rawTrainResult is a collection consisting of pattern label and raw values (before clip) of each target for each pattern, and is used for binary output in discrete Neugent;

rawValidationResult is a collection consisting of pattern label and raw values (before clip) of each target for each pattern used for validation, and is used for binary output in discrete Neugent;

originalTrainOutput is a collection consisting of pattern label and original values of each target for each pattern used for training;

originalValidationOutput is a collection consisting of pattern label and original values of each target for each pattern used for validation; and

neugentModel is a shortcut to the model that uses the ValueNeugentTrainResult object.

The ValueNeugentConsultResult class contains the results from the ValuePredictNeugent consult method, and may include the following fields (FIG. 4B):

consultError indicates for each output on the OFldNList of the Neugent object a corresponding error, and is empty if the target value is not included on the consult data source;

consultQualityScore indicates for each output on the OFldNList of the Neugent object a corresponding quality score, and is empty if the target value is not included on the consult data source;

consultResult is a collection consisting of pattern label and predict values of each output for each pattern;

originalConsultOutput is a collection consisting of pattern label and original output values for each pattern;

rawConsultResult is a collection consisting of pattern label and binary output values for each pattern, and is used for binary output in discrete Neugent; and

neugentObject is a shortcut to a model that uses the ValueNeugentTrainResult object.

Class diagrams for additional exemplary embodiments are shown in FIGS. 5A-5K, 6A-6K, 7A-7G, 8A-8K, 9A-9C and 10A-10F. Similarly named field have similar functionality as described above. In the interest of clarity, a description of the fields in the additional exemplary embodiments is omitted.

The above specific embodiments are illustrative, and many variations can be introduced on these embodiments without departing from the spirit of the disclosure or from the scope of the appended claims. Elements and/or features of different illustrative embodiments may be combined with each other and/or substituted for each other within the

scope of this disclosure and appended claims.

For example, although some embodiments described herein use a combination of ClusteringNeugent, DecisionNeugent, EventPredictNeugent, TextClusteringNeugent and ValuePredictNeugent methodologies, the matter recited in the appended claims may be practiced a selected subset of these Neugents, with or without other Neugents technologies which use clustering, neural net, decision tree and/or other predictive modeling methodologies.

Additional variations may be apparent to one of ordinary skill in the art from reading the following U.S. provisional applications Nos. 60/374,064, 60/374,020, f60/374,024, 60/374,041, 60/373,977 and 60/373,780, each filed April 19, 2002.